```
EEEEEEEEEEEEEEE  RRRRRRRRRRR     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRRR    FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRRR    FFFFFFFFFFFFFFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRRR     FFFFFFFFFFFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
```

```
LL             AAAAAA    BBBBBBBB    EEEEEEEEEE  LL
LL             AAAAAA    BBBBBBBB    EEEEEEEEEE  LL
LL           AA      AA  BB      BB  EE          LL
LL           AA      AA  BB      BB  EE          LL
LL           AA      AA  BB      BB  EE          LL
LL           AA      AA  BB      BB  EE          LL
LL           AA      AA  BBBBBBBB    EEEEEEEE    LL
LL           AA      AA  BBBBBBBB    EEEEEEEE    LL
LL           AAAAAAAAAA  BB      BB  EE          LL
LL           AAAAAAAAAA  BB      BB  EE          LL
LL           AA      AA  BB      BB  EE          LL              ....
LL           AA      AA  BB      BB  EE          LL              ....
LLLLLLLLLL   AA      AA  BBBBBBBB    EEEEEEEEEE  LLLLLLLLLL      ....
LLLLLLLLLL   AA      AA  BBBBBBBB    EEEEEEEEEE  LLLLLLLLLL      ....


LL             IIIIII      SSSSSSSS
LL             IIIIII      SSSSSSSS
LL               II      SS
LL               II      SS
LL               II      SS
LL               II      SS
LL               II        SSSSSS
LL               II        SSSSSS
LL               II              SS
LL               II              SS
LL               II              SS
LL               II              SS
LLLLLLLLLL     IIIIII    SSSSSSSS
LLLLLLLLLL     IIIIII    SSSSSSSS
```

```
0001    C
0002    C Version:      'V04-000'
0003    C
0004    C*****************************************************************************
0005    C*                                                                           *
0006    C*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0007    C*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0008    C*  ALL RIGHTS RESERVED.                                                     *
0009    C*                                                                           *
0010    C*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0011    C*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0012    C*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0013    C*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0014    C*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0015    C*  TRANSFERRED.                                                             *
0016    C*                                                                           *
0017    C*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0018    C*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0019    C*  CORPORATION.                                                             *
0020    C*                                                                           *
0021    C*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0022    C*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0023    C*                                                                           *
0024    C*                                                                           *
0025    C*****************************************************************************
0026    C
0027    C
0028    C       AUTHOR  BRIAN PORTER            CREATION DATE   22-MAY-1980
0029    C
0030    C
0031
0032    c++
0033    c       Modified by:
0034    c
0035    C       V03-002 SAR0075         Sharon A. Reynolds,     20-Jun-1983
0036    C               Changed the carriage control in the 'format' statements
0037    C               for use with ERF.
0038    C
0039    C       V03-001 SAR0021         Sharon A. Reynolds,     4-May-1983
0040    C               Made label_out a subroutine. Modified 'label_out' so that
0041    C               it calls 'get_queue_info' to get root flink.
0042    C
0043    c       v02-004 BP0004          Brian Porter,           23-JAN-1982
0044    c               Made label list alphabetical.
0045    c
0046    c       v02-003 BP0003          Brian Porter,           16-NOV-1981
0047    c               Added control-o code.
0048    c
0049    c       v02-002 BP0002          Brian Porter,           06-MAY-1981
0050    c               Added an extra linefeed to the 'volume' herald.  Removed
0051    c               RETURN 1 argument.
0052    c
0053    c       v02-001 BP0001          Brian Porter,           27-JAN-1981
0054    c               Added code to put unit's in ascending order.  Added code
0055    c               to reprint label heading for devices of different names.
0056    c
0057    c       Functional description:
```

```
0058     c
0059     c       This routine maintains a four dimensional list that keeps
0060     c       track of the errors that occur on unique volume labels
0061     c       as they traverse various devices.
0062     c
0063     c       The first dimension has absolute linkage and the following format.
0064     c
0065     c       +-----------------------------------+
0066     c       I             flink1                I
0067     c       +-----------------------------------+
0068     c       I             blink1                I
0069     c       +-----------------------------------+
0070     c       I           logging SID             I
0071     c       +-----------------------------------+
0072     c       I         root label flink          I
0073     c       +-----------------------------------+
0074     c       I         root label blink          I
0075     c       +-----------------------------------+
0076     c       I         label entry count         I
0077     c       +-----------------------------------+
0078     c
0079     c       The second dimension has absolute linkage and the following format.
0080     c
0081     c       +-------------------------+
0082     c       I         flink2          I
0083     c       +-------------------------+
0084     c       I         blink2          I
0085     c       +-------------------------+
0086     c       I                         I
0087     c       +--                     --+
0088     c       I  12 byte label field    I
0089     c       +--                     --+
0090     c       I                         I
0091     c       +-------------------------+
0092     c       I     root name flink     I
0093     c       +-------------------------+
0094     c       I     root name blink     I
0095     c       +-------------------------+
0096     c       I     name entry count    I
0097     c       +-------------------------+
0098     c
0099     c       The third dimension has absolute linkage and the following format.
0100     c
0101     c       +---------------------------+
0102     c       I          flink3           I
0103     c       +---------------------------+
0104     c       I          blink3           I
0105     c       +---------------------------+
0106     c       I                           I
0107     c       +--                       --+
0108     c       I       16 byte name        I
0109     c       +--                       --+
0110     c       I          field            I
0111     c       +--                       --+
0112     c       I                           I
0113     c       +---------------------------+
0114     c       I      root unit flink      I
```

```
0115     c        +----------------------------+
0116     c        I      root unit blink       I
0117     c        +----------------------------+
0118     c        I      unit entry count      I
0119     c        +----------------------------+
0120     c
0121     c        The fourth dimension has absolute linkage and the following format
0122     c
0123     c        +----------------------------+
0124     c        I           flink4           I
0125     c        +----------------------------+
0126     c        I           blink4           I
0127     c        +----------------------------+
0128     c        I        ucb unit number     I
0129     c        +----------------------------+
0130     c        I   mount operation count    I
0131     c        +----------------------------+
0132     c        I   mount error count        I
0133     c        +----------------------------+
0134     c        I dismount operation count   I
0135     c        +----------------------------+
0136     c        I  dismount error count      I
0137     c        +----------------------------+
0138     c        I      mount count           I
0139     c        +----------------------------+
0140     c        I      mounted flag          I
0141     c        +----------------------------+
0142     c        I   mount before dismount    I
0143     c        +----------------------------+
0144     c        I  last mount operation cnt  I
0145     c        +----------------------------+
0146     c        I  last mount error count    I
0147     c        +----------------------------+
0148     c
0149     c        Subroutine LABEL is called whenever mount/dismount or device
0150     c        error/timeout entries are encountered.
0151     c
0152     c        If the entry type is mount then an a search is made for a list entry
0153     c        where the device name,  volume label and unit number are the same as
0154     c        the error log entry.  If found then the counters for that list entry
0155     c        are updated, otherwise a new list entry is created.
0156     c        If the entry type is dismount then a search is made for a list entry
0157     c        that corresponds to this device name,volume label and unit number.
0158     c
0159     c        To overcome the problem of random mounts and dismounts of the same
0160     c        volume label on a particular drive two boolean variables and
0161     c        two counters are used.  The boolean variables are used to synchronize
0162     c        correctness of mount/dismount sequences, the counters are used to
0163     c        store values of operation and error counts for individual units for
0164     c        particular volume labels.
0165     c**
0166     c--
0167
0168
0169              subroutine label (entrance,search_sid,search_name_length,
0170              1 search_name_string,search_unit,search_label,operation_count,
0171              1 error_count)
```

```
0172
0173            byte            lun
0174
0175            integer*4       buffer0(2)
0176            integer*4       buffer1(6)
0177            integer*4       buffer2(8)
0178            integer*4       buffer3(9)
0179            integer*4       buffer4(12)
0180            integer*4       root_logging_sid_flink
0181            integer*4       root_logging_sid_blink
0182            Integer*4       Root_flink
0183            Integer*4       Sid_count
0184            Integer*4       Label_count
0185            Integer*4       Name_count
0186            Integer*4       Unit_count
0187            Integer*4       Logging_sid_entry_count
0188            Integer*4       Label_entry_count
0189            Integer*4       Name_entry_count
0190            Integer*4       Unit_entry_count
0191
0192            equivalence     (buffer0(1),root_logging_sid_flink)
0193            equivalence     (buffer0(2),root_logging_sid_blink)
0194
0195            integer*4       flink1
0196            integer*4       blink1
0197            integer*4       logging_sid
0198            integer*4       root_label_flink
0199            integer*4       root_label_blink
0200
0201            equivalence     (buffer1(1),flink1)
0202            equivalence     (buffer1(2),blink1)
0203            equivalence     (buffer1(3),logging_sid)
0204            equivalence     (buffer1(4),root_label_flink)
0205            equivalence     (buffer1(5),root_label_blink)
0206            equivalence     (buffer1(6),label_entry_count)
0207
0208            integer*4       flink2
0209            integer*4       blink2
0210
0211            byte            label_array(12)
0212
0213            character*12    label_string
0214
0215            integer*4       root_name_flink
0216            integer*4       root_name_blink
0217
0218            equivalence     (buffer2(1),flink2)
0219            equivalence     (buffer2(2),blink2)
0220            equivalence     (buffer2(3),label_array)
0221            equivalence     (label_array,label_string)
0222            equivalence     (buffer2(6),root_name_flink)
0223            equivalence     (buffer2(7),root_name_blink)
0224            equivalence     (buffer2(8),name_entry_count)
0225
0226            integer*4       flink3
0227            integer*4       blink3
0228
```

```
0229            byte            name_array(16)
0230            byte            name_length
0231
0232            character*15    name_string
0233
0234            integer*4       root_unit_flink
0235            integer*4       root_unit_blink
0236
0237            equivalence     (buffer3(1),flink3)
0238            equivalence     (buffer3(2),blink3)
0239            equivalence     (buffer3(3),name_array)
0240            equivalence     (name_array,name_length)
0241            equivalence     (name_array(2),name_string)
0242            equivalence     (buffer3(7),root_unit_flink)
0243            equivalence     (buffer3(8),root_unit_blink)
0244            equivalence     (buffer3(9),unit_entry_count)
0245
0246            integer*4       flink4
0247            integer*4       blink4
0248            integer*4       ucb_unit_number
0249            integer*4       ucb_mount_operation_count
0250            integer*4       ucb_mount_error_count
0251            integer*4       ucb_dismount_operation_count
0252            integer*4       ucb_dismount_error_count
0253            integer*4       sye_mount_count
0254
0255            logical*4       mounted
0256            logical*4       mount_before_dismount
0257
0258            integer*4       last_valid_mount_opration_count
0259            integer*4       last_valid_mount_error_count
0260
0261            equivalence     (buffer4(1),flink4)
0262            equivalence     (buffer4(2),blink4)
0263            equivalence     (buffer4(3),ucb_unit_number)
0264            equivalence     (buffer4(4),ucb_mount_operation_count)
0265            equivalence     (buffer4(5),ucb_mount_error_count)
0266            equivalence     (buffer4(6),ucb_dismount_operation_count)
0267            equivalence     (buffer4(7),ucb_dismount_error_count)
0268            equivalence     (buffer4(8),sye_mount_count)
0269            equivalence     (buffer4(9),mounted)
0270            equivalence     (buffer4(10),mount_before_dismount)
0271            equivalence     (buffer4(11),last_valid_mount_opration_count)
0272            equivalence     (buffer4(12),last_valid_mount_error_count)
0273
0274            integer*4       logging_sid_entry_address
0275            integer*4       label_entry_address
0276            integer*4       name_entry_address
0277            integer*4       unit_entry_address
0278            integer*4       search_sid
0279            integer*4       entrance
0280
0281            integer*2       search_unit
0282
0283            character*15    search_name_string
0284            character*15    search_name
0285
```

```
0286            byte            search_name_length
0287
0288            character*12    search_label
0289
0290            logical*1       lib$get_vm
0291
0292            integer*4       lib$extzv
0293            integer*4       compress4
0294            integer*4       operation_count
0295            integer*4       error_count
0296            integer*4       label_operation_count
0297            integer*4       label_error_count
0298
0299            logical*1       label_herald_printed
0300            logical*1       sid_herald_printed
0301
0302            byte            operation_width
0303            byte            error_width
0304            byte            mount_width
0305
0306            integer*4       insert_blink
0307
0308            character*15    previous_name_string
0309
0310
0311
0312
0313            call movc5 (%val(search_name_length),%ref(search_name_string),%val(42),
0314            1 %val(15),%ref(search_name))
0315
0316            logging_sid_entry_address = root_logging_sid_flink
0317
0318            do 100,i = 1,logging_sid_entry_count
0319
0320            call movc3 (%val(24),%val(logging_sid_entry_address),buffer1)
0321
0322    5       if (logging_sid .eq. search_sid) then
0323
0324            label_entry_address = root_label_flink
0325
0326            do 90,j = 1,label_entry_count
0327
0328            call movc3 (%val(32),%val(label_entry_address),buffer2)
0329
0330    8       if (search_label .eq. label_string) then
0331
0332            name_entry_address = root_name_flink
0333
0334            do 80,k = 1,name_entry_count
0335
0336            call movc3 (%val(36),%val(name_entry_address),buffer3)
0337
0338    10      if (search_name .eq. name_string) then
0339
0340            unit_entry_address = root_unit_flink
0341
0342            do 60,l = 1,unit_entry_count
```

```
0343
0344            call movc3 (%val(48),%val(unit_entry_address),buffer4)
0345
0346    15      if (search_unit .eq. ucb_unit_number) then
0347
0348            goto (300,400) entrance
0349
0350            return
0351            endif
0352
0353            insert_blink = blink4
0354
0355            if (ucb_unit_number .gt. search_unit) goto 65
0356
0357            unit_entry_address = flink4
0358
0359    60      continue
0360
0361            insert_blink = root_unit_blink
0362
0363    65      if (entrance .eq. 2) return
0364
0365            call movc5 (%val(0),,%val(0),%val(48),buffer4)
0366
0367            if(lib$get_vm(((48+7)/8)*8,unit_entry_address)) then
0368
0369            call insque (%val(unit_entry_address),%val(insert_blink))
0370
0371            ucb_unit_number = search_unit
0372
0373            unit_entry_count = unit_entry_count + 1
0374
0375            call movl (unit_entry_count,%val(name_entry_address + 32))
0376
0377            goto 15
0378            endif
0379
0380            return
0381            endif
0382
0383            name_entry_address = flink3
0384
0385    80      continue
0386
0387            if (entrance .eq. 2) return
0388
0389            call movc5 (%val(0),,%val(0),%val(36),buffer3)
0390
0391            if (lib$get_vm(((36+7)/8)*8,name_entry_address)) then
0392
0393            call insque (%val(name_entry_address),%val(root_name_blink))
0394
0395            name_length = search_name_length
0396
0397            name_string = search_name
0398
0399            root_unit_flink = name_entry_address + 24
```

```
0400
0401            root_unit_blink = root_unit_flink
0402
0403            call movc3 (%val(28),name_length,%val(name_entry_address + 8))
0404
0405            name_entry_count = name_entry_count + 1
0406
0407            call movl (name_entry_count,%val(label_entry_address + 28))
0408
0409            goto 10
0410            endif
0411
0412            return
0413            endif
0414
0415            insert_blink = blink2
0416
0417            do 85,m = 1,12
0418
0419            if (ichar(label_string(m:m)) - ichar(search_label(m:m))) 87,85,95
0420
0421    85      continue
0422
0423    87      label_entry_address = flink2
0424
0425    90      continue
0426
0427            insert_blink = root_label_blink
0428
0429    95      if (entrance .eq. 2) return
0430
0431            call movc5 (%val(0),,%val(0),%val(32),buffer2)
0432
0433            if (lib$get_vm(((32+7)/8)*8,label_entry_address)) then
0434
0435            call insque (%val(label_entry_address),%val(insert_blink))
0436
0437            root_name_flink = label_entry_address + 20
0438
0439            root_name_blink = root_name_flink
0440
0441            label_string = search_label
0442
0443            call movc3 (%val(24),%ref(label_string),%val(label_entry_address + 8))
0444
0445            label_entry_count = label_entry_count + 1
0446
0447            call movl (label_entry_count,%val(logging_sid_entry_address + 20))
0448
0449            goto 8
0450            endif
0451
0452            return
0453            endif
0454
0455            logging_sid_entry_address = flink1
0456
```

```
0457     100      continue
0458
0459              if (entrance .eq. 2) return
0460
0461              call movc5 (%val(0),,%val(0),%val(24),buffer1)
0462
0463              if (lib$get_vm(((24+7)/8)*8,logging_sid_entry_address)) then
0464
0465              if (logging_sid_entry_count .eq. 0) then
0466
0467              root_logging_sid_flink = %loc(root_logging_sid_flink)
0468
0469              root_logging_sid_blink = %loc(root_logging_sid_flink)
0470              endif
0471
0472              call insque (%val(logging_sid_entry_address),
0473             1 %val(root_logging_sid_blink))
0474
0475              logging_sid = search_sid
0476
0477              root_label_flink = logging_sid_entry_address + 12
0478
0479              root_label_blink = root_label_flink
0480
0481              logging_sid_entry_count = logging_sid_entry_count + 1
0482
0483              call movc3 (%val(16),logging_sid,%val(logging_sid_entry_address + 8))
0484
0485              goto 5
0486              endif
0487
0488              return
0489
0490     c
0491     c        action routine for MOUNT VOLUME calls
0492     c
0493
0494     300      continue
0495
0496              last_valid_mount_opration_count = operation_count
0497
0498              last_valid_mount_error_count = error_count
0499
0500              mounted = .true.
0501
0502              call movc3 (%val(40),ucb_unit_number,%val(unit_entry_address + 8))
0503
0504              return
0505
0506     c
0507     c        action routine for DISMOUNT VOLUME calls
0508     c
0509
0510     400      continue
0511
0512              if (mounted) then
0513
```

```
0514            if (operation_count .ge. last_valid_mount_opration_count
0515            1 .and.
0516            1 error_count .ge. last_valid_mount_error_count) then
0517
0518            ucb_mount_operation_count = ucb_mount_operation_count +
0519            1 last_valid_mount_opration_count
0520
0521            ucb_mount_error_count = ucb_mount_error_count +
0522            1 last_valid_mount_error_count
0523
0524            ucb_dismount_operation_count = ucb_dismount_operation_count +
0525            1 operation_count
0526
0527            ucb_dismount_error_count = ucb_dismount_error_count + error_count
0528
0529            sye_mount_count = sye_mount_count + 1
0530
0531            mounted = .false.
0532
0533            call movc3 (%val(40),ucb_unit_number,%val(unit_entry_address + 8))
0534            endif
0535            endif
0536
0537            return
0538
0539
0540            Entry GET_QUEUE_INFO (ROOT_FLINK,SID_COUNT,
0541            1 LABEL_COUNT,NAME_COUNT,UNIT_COUNT)
0542
0543            Root_flink = root_logging_sid_flink
0544            Sid_count = logging_sid_entry_count
0545            Label_count = label_entry_count
0546            Name_count = name_entry_count
0547            Unit_count = unit_entry_count
0548            Return
0549
0550
0551            End
```

PROGRAM SECTIONS

	Name	Bytes	Attributes
0	$CODE	933	PIC CON REL LCL SHR EXE RD NOWRT LONG
1	$PDATA	16	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2	$LOCAL	620	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
	Total Space Allocated	1569	

ENTRY POINTS

Address	Type	Name	Address	Type	Name
0-00000383		GET_QUEUE_INFO	0-00000000		LABEL

VARIABLES

Address	Type	Name	Address	Type	Name
2-00000078	I*4	BLINK1	2-00000058	I*4	BLINK2
2-00000034	I*4	BLINK3	2-00000004	I*4	BLINK4
2-000000D0	I*4	COMPRESS4	AP-00000004@	I*4	ENTRANCE
AP-00000020@	I*4	ERROR_COUNT	2-000000A7	L*1	ERROR_WIDTH
2-00000074	I*4	FLINK1	2-00000054	I*4	FLINK2
2-00000030	I*4	FLINK3	2-00000000	I*4	FLINK4
2-000000E0	I*4	I	2-000000DC	I*4	INSERT_BLINK
2-000000E4	I*4	J	2-000000E8	I*4	K
2-000000EC	I*4	L	AP-0000000C@	I*4	LABEL_COUNT
2-000000C0	I*4	LABEL_ENTRY_ADDRESS	2-00000088	I*4	LABEL_ENTRY_COUNT
2-000000D8	I*4	LABEL_ERROR_COUNT	2-000000A4	L*1	LABEL_HERALD_PRINTED
2-000000D4	I*4	LABEL_OPERATION_COUNT	2-0000005C	CHAR	LABEL_STRING
2-0000002C	I*4	LAST_VALID_MOUNT_ERROR_COUNT	2-00000028	I*4	LAST_VALID_MOUNT_OPRATION_COUNT
2-000000CC	I*4	LIB$EXTZV	2-0000007C	I*4	LOGGING_SID
2-000000BC	I*4	LOGGING_SID_ENTRY_ADDRESS	2-000000B8	I*4	LOGGING_SID_ENTRY_COUNT
2-00000094	L*1	LUN	2-000000F0	I*4	M
2-00000020	L*4	MOUNTED	2-00000024	L*4	MOUNT_BEFORE_DISMOUNT
2-000000A8	L*1	MOUNT_WIDTH	AP-00000010@	I*4	NAME_COUNT
2-000000C4	I*4	NAME_ENTRY_ADDRESS	2-00000070	I*4	NAME_ENTRY_COUNT
2-00000038	L*1	NAME_LENGTH	2-00000039	CHAR	NAME_STRING
AP-0000001C@	I*4	OPERATION_COUNT	2-000000A6	L*1	OPERATION_WIDTH
2-000000A9	CHAR	PREVIOUS_NAME_STRING	AP-00000004@	I*4	ROOT_FLINK
2-00000084	I*4	ROOT_LABEL_BLINK	2-00000080	I*4	ROOT_LABEL_FLINK
2-00000090	I*4	ROOT_LOGGING_SID_BLINK	2-0000008C	I*4	ROOT_LOGGING_SID_FLINK
2-0000006C	I*4	ROOT_NAME_BLINK	2-00000068	I*4	ROOT_NAME_FLINK
2-0000004C	I*4	ROOT_UNIT_BLINK	2-00000048	I*4	ROOT_UNIT_FLINK
AP-00000018@	CHAR	SEARCH_LABEL	2-00000095	CHAR	SEARCH_NAME
AP-0000000C@	L*1	SEARCH_NAME_LENGTH	AP-00000010@	CHAR	SEARCH_NAME_STRING
AP-00000008@	I*4	SEARCH_SID	AP-00000014@	I*2	SEARCH_UNIT
AP-00000008@	I*4	SID_COUNT	2-000000A5	L*1	SID_HERALD_PRINTED
2-0000001C	I*4	SYE_MOUNT_COUNT	2-00000018	I*4	UCB_DISMOUNT_ERROR_COUNT
2-00000014	I*4	UCB_DISMOUNT_OPERATION_COUNT	2-00000010	I*4	UCB_MOUNT_ERROR_COUNT
2-0000000C	I*4	UCB_MOUNT_OPERATION_COUNT	2-00000008	I*4	UCB_UNIT_NUMBER
AP-00000014@	I*4	UNIT_COUNT	2-000000C8	I*4	UNIT_ENTRY_ADDRESS
2-00000050	I*4	UNIT_ENTRY_COUNT			

ARRAYS

Address	Type	Name	Bytes	Dimensions
2-0000008C	I*4	BUFFER0	8	(2)
2-00000074	I*4	BUFFER1	24	(6)
2-00000054	I*4	BUFFER2	32	(8)
2-00000030	I*4	BUFFER3	36	(9)
2-00000000	I*4	BUFFER4	48	(12)
2-0000005C	L*1	LABEL_ARRAY	12	(12)
2-00000038	L*1	NAME_ARRAY	16	(16)

LABELS

Address	Label	Address	Label	Address	Label	Address	Label	Address	Label	Address	Label
0-00000051	5	0-00000082	8	0-000000B6	10	0-000000E7	15	**	60	0-0000011B	65
**	80	**	85	0-00000219	87	**	90	0-0000022C	95	**	100
0-00000313	300	0-00000338	400								

FUNCTIONS AND SUBROUTINES REFERENCED

Type	Name	Type	Name	Type	Name	Type	Name	Type	Name
	INSQUE	L*1	iB$GET_VM		MOVC3		MOVC5		MOVL

```
0001
0002
```

```
0003            Subroutine LABEL_OUT (lun)
0004
0005    C++
0006    C       Functional Description:
0007    C
0008    C       This module handles the output of the volume summary information.
0009    C
0010    C--
0011
0012            byte            lun
0013
0014            integer*4       buffer0(2)
0015            integer*4       buffer1(6)
0016            integer*4       buffer2(8)
0017            integer*4       buffer3(9)
0018            integer*4       buffer4(12)
0019            integer*4       root_logging_sid_flink
0020            integer*4       root_logging_sid_blink
0021
0022            equivalence     (buffer0(1),root_logging_sid_flink)
0023            equivalence     (buffer0(2),root_logging_sid_blink)
0024
0025            integer*4       flink1
0026            integer*4       blink1
0027            integer*4       logging_sid
0028            integer*4       root_label_flink
0029            integer*4       root_label_blink
0030
0031            equivalence     (buffer1(1),flink1)
0032            equivalence     (buffer1(2),blink1)
0033            equivalence     (buffer1(3),logging_sid)
0034            equivalence     (buffer1(4),root_label_flink)
0035            equivalence     (buffer1(5),root_label_blink)
0036            equivalence     (buffer1(6),label_entry_count)
0037
0038            integer*4       flink2
0039            integer*4       blink2
0040
0041            byte            label_array(12)
0042
0043            character*12    label_string
0044
0045            integer*4       root_name_flink
0046            integer*4       root_name_blink
0047
0048            equivalence     (buffer2(1),flink2)
0049            equivalence     (buffer2(2),blink2)
0050            equivalence     (buffer2(3),label_array)
0051            equivalence     (label_array,label_string)
0052            equivalence     (buffer2(6),root_name_flink)
0053            equivalence     (buffer2(7),root_name_blink)
0054            equivalence     (buffer2(8),name_entry_count)
0055
0056            integer*4       flink3
0057            integer*4       blink3
0058
0059            byte            name_array(16)
```

```
0060            byte            name_length
0061
0062            character*15    name_string
0063
0064            integer*4       root_unit_flink
0065            integer*4       root_unit_blink
0066
0067            equivalence     (buffer3(1),flink3)
0068            equivalence     (buffer3(2),blink3)
0069            equivalence     (buffer3(3),name_array)
0070            equivalence     (name_array,name_length)
0071            equivalence     (name_array(2),name_string)
0072            equivalence     (buffer3(7),root_unit_flink)
0073            equivalence     (buffer3(8),root_unit_blink)
0074            equivalence     (buffer3(9),unit_entry_count)
0075
0076            integer*4       flink4
0077            integer*4       blink4
0078            integer*4       sye_mount_count
0079            integer*4       ucb_dismount_operation_count
0080            integer*4       ucb_dismount_error_count
0081            integer*4       ucb_mount_operation_count
0082            integer*4       ucb_mount_error_count
0083            integer*4       ucb_unit_number
0084
0085            logical*4       mount_before_dismount
0086            logical*4       mounted
0087
0088            integer*4       last_valid_mount_error_count
0089            integer*4       last_valid_mount_opration_count
0090
0091            equivalence     (buffer4(1),flink4)
0092            equivalence     (buffer4(2),blink4)
0093            equivalence     (buffer4(3),ucb_unit_number)
0094            equivalence     (buffer4(4),ucb_mount_operation_count)
0095            equivalence     (buffer4(5),ucb_mount_error_count)
0096            equivalence     (buffer4(6),ucb_dismount_operation_count)
0097            equivalence     (buffer4(7),ucb_dismount_error_count)
0098            equivalence     (buffer4(8),sye_mount_count)
0099            equivalence     (buffer4(9),mounted)
0100            equivalence     (buffer4(10),mount_before_dismount)
0101            equivalence     (buffer4(11),last_valid_mount_opration_count)
0102            equivalence     (buffer4(12),last_valid_mount_error_count)
0103
0104            byte            error_width
0105            byte            mount_width
0106            byte            operation_width
0107            byte            search_name_length
0108
0109            logical*1       lib$get_vm
0110            logical*1       label_herald_printed
0111            logical*1       sid_herald_printed
0112
0113            integer*2       search_unit
0114
0115            integer*4       compress4
0116            integer*4       entrance
```

```
0117            integer*4       error_count
0118            integer*4       insert_blink
0119            integer*4       label_entry_address
0120            integer*4       label_operation_count
0121            integer*4       label_error_count
0122            integer*4       logging_sid_entry_address
0123            integer*4       libsextzv
0124            integer*4       name_entry_address
0125            integer*4       operation_count
0126            integer*4       search_sid
0127            integer*4       unit_entry_address
0128
0129            character*12    search_label
0130            character*15    search_name_string
0131            character*15    search_name
0132            character*15    previous_name_string
0133
0134            Integer*4       Logging_sid_entry_count
0135            Integer*4       Label_entry_count
0136            Integer*4       Name_entry_count
0137            Integer*4       Unit_entry_count
0138
0139    C
0140    C Get the root flink for the volume information queue.
0141    C
0142            Call GET_QUEUE_INFO (root_logging_sid_flink,logging_sid_entry_count,
0143            1 label_entry_count,name_entry_count,unit_entry_count)
0144
0145            logging_sid_entry_address = root_logging_sid_flink
0146
0147            do 200,i = 1,logging_sid_entry_count
0148
0149            call movc3 (%val(24),%val(logging_sid_entry_address),buffer1)
0150
0151            sid_herald_printed = .false.
0152
0153            label_entry_address = root_label_flink
0154
0155            do 195,j = 1,label_entry_count
0156
0157            label_herald_printed = .false.
0158
0159            call movc3 (%val(32),%val(label_entry_address),buffer2)
0160
0161            name_entry_address = root_name_flink
0162
0163            do 190,k = 1,name_entry_count
0164
0165            call movc3 (%val(36),%val(name_entry_address),buffer3)
0166
0167            unit_entry_address = root_unit_flink
0168
0169            do 185,l = 1,unit_entry_count
0170
0171            call movc3 (%val(48),%val(unit_entry_address),buffer4)
0172
0173            if (sye_mount_count .ne. 0) then
```

```
0174
0175            if (.not. sid_herald_printed) then
0176
0177    c       call set_rab$v_cco
0178
0179            call frctof (lun)
0180
0181            call linchk (lun,3)
0182
0183            write(lun,105) logging_sid
0184    105     format(/' ','VOLUME LABEL(S) LOGGED BY SID ',z8.8,//,
0185           1 t34,'QIO($)',t44,'ERROR(S)',t54,'MOUNT(S)')
0186
0187            sid_herald_printed = .true.
0188            endif
0189
0190            if (name_string .ne. previous_name_string)
0191           1 label_herald_printed = .false.
0192
0193            if (.not. label_herald_printed) then
0194
0195            call linchk (lun,3)
0196
0197            write(lun,110) label_string
0198    110     format(/' ',t8,'LABEL -- ',a,/)
0199
0200            label_herald_printed = .true.
0201            endif
0202
0203            label_operation_count = ucb_dismount_operation_count -
0204           1 ucb_mount_operation_count
0205
0206            label_error_count = ucb_dismount_error_count - ucb_mount_error_count
0207
0208            operation_width = compress4 (label_operation_count)
0209
0210            error_width = compress4 (label_error_count)
0211
0212            mount_width = compress4 (sys_mount_count)
0213
0214            call linchk (lun,1)
0215
0216            write(lun,115) name_string(1:name_length),ucb_unit_number,
0217           1 label_operation_count,label_error_count,sys_mount_count
0218    115     format(' ',t8,' ',a<name_length>,i<compress4 (ucb_unit_number)>,':',
0219           1 t<40 - operation_width>,i<operation_width>,'.',
0220           1 t<52 - error_width>,i<error_width>,'.'
0221           1 t<62 - mount_width>,i<mount_width>,'.')
0222            endif
0223
0224            unit_entry_address = flink4
0225
0226            previous_name_string = name_string
0227
0228    185     continue
0229
0230            name_entry_address = flink3
```

```
0231
0232     190      continue
0233
0234              label_entry_address = flink2
0235
0236     195      continue
0237
0238              logging_sid_entry_address = flink1
0239
0240     200      continue
0241
0242              return
0243              end
```

PROGRAM SECTIONS

	Name	Bytes	Attributes
0	$CODE	676	PIC CON REL LCL SHR EXE RD NOWRT LONG
1	$PDATA	173	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2	$LOCAL	464	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
	Total Space Allocated	1313	

ENTRY POINTS

Address	Type	Name
0-00000000		LABEL_OUT

VARIABLES

Address	Type	Name	Address	Type	Name
2-00000078	I*4	BLINK1	2-00000058	I*4	BLINK2
2-00000034	I*4	BLINK3	2-00000004	I*4	BLINK4
2-000000D8	I*4	ENTRANCE	2-000000DC	I*4	ERROR_COUNT
2-00000094	L*1	ERROR_WIDTH	2-00000074	I*4	FLINK1
2-00000054	I*4	FLINK2	2-00000030	I*4	FLINK3
2-00000000	I*4	FLINK4	2-0000010C	I*4	I
2-000000E0	I*4	INSERT_BLINK	2-00000110	I*4	J
2-00000114	I*4	K	2-00000118	I*4	L
2-000000E4	I*4	LABEL_ENTRY_ADDRESS	2-00000088	I*4	LABEL_ENTRY_COUNT
2-000000EC	I*4	LABEL_ERROR_COUNT	2-00000099	L*1	LABEL_HERALD_PRINTED
2-000000E8	I*4	LABEL_OPERATION_COUNT	2-0000005C	CHAR	LABEL_STRING
2-0000002C	I*4	LAST_VALID_MOUNT_ERROR_COUNT	2-00000028	I*4	LAST_VALID_MOUNT_OPRATION_COUNT
2-000000F4	I*4	LIB$EXTZV	2-00000098	L*1	LIB$GET_VM
2-0000007C	I*4	LOGGING_SID	2-000000F0	I*4	LOGGING_SID_ENTRY_ADDRESS
2-00000108	I*4	LOGGING_SID_ENTRY_COUNT	AP-00000004@	L*1	LUN
2-00000020	L*4	MOUNTED	2-00000024	L*4	MOUNT_BEFORE_DISMOUNT
2-00000095	L*1	MOUNT_WIDTH	2-000000F8	I*4	NAME_ENTRY_ADDRESS
2-00000070	I*4	NAME_ENTRY_COUNT	2-00000038	L*1	NAME_LENGTH

```
  2-00000039  CHAR NAME_STRING                      2-000000FC  I*4  OPERATION_COUNT
  2-00000096  L*1  OPERATION_WIDTH                  2-000000C5  CHAR PREVIOUS_NAME_STRING
  2-00000084  I*4  ROOT_LABEL_BLINK                 2-00000080  I*4  ROOT_LABEL_FLINK
  2-00000090  I*4  ROOT_LOGGING_SID_BLINK           2-0000008C  I*4  ROOT_LOGGING_SID_FLINK
  2-0000006C  I*4  ROOT_NAME_BLINK                  2-00000068  I*4  ROOT_NAME_FLINK
  2-0000004C  I*4  ROOT_UNIT_BLINK                  2-00000048  I*4  ROOT_UNIT_FLINK
  2-0000009B  CHAR SEARCH_LABEL                     2-000000B6  CHAR SEARCH_NAME
  2-00000097  L*1  SEARCH_NAME_LENGTH               2-000000A7  CHAR SEARCH_NAME_STRING
  2-00000100  I*4  SEARCH_SID                       2-000000D4  I*2  SEARCH_UNIT
  2-0000009A  L*1  SID_HERALD_PRINTED               2-0000001C  I*4  SYE_MOUNT_COUNT
  2-00000018  I*4  UCB_DISMOUNT_ERROR_COUNT         2-00000014  I*4  UCB_DISMOUNT_OPERATION_COUNT
  2-00000010  I*4  UCB_MOUNT_ERROR_COUNT            2-0000000C  I*4  UCB_MOUNT_OPERATION_COUNT
  2-00000008  I*4  UCB_UNIT_NUMBER                  2-00000104  I*4  UNIT_ENTRY_ADDRESS
  2-00000050  I*4  UNIT_ENTRY_COUNT

ARRAYS

    Address  Type  Name                 Bytes  Dimensions

  2-0000008C  I*4  BUFFER0                  8  (2)
  2-00000074  I*4  BUFFER1                 24  (6)
  2-00000054  I*4  BUFFER2                 32  (8)
  2-00000030  I*4  BUFFER3                 36  (9)
  2-00000000  I*4  BUFFER4                 48  (12)
  2-0000005C  L*1  LABEL_ARRAY             12  (12)
  2-00000038  L*1  NAME_ARRAY              16  (16)

LABELS

    Address   Label      Address   Label      Address   Label      Address   Label      Address   Label      Address   Label
  1-00000008  105'     1-00000054  110'     1-00000068  115'        **       185         **       190         **       195
       **     200

FUNCTIONS AND SUBROUTINES REFERENCED

  Type  Name                              Type  Name                              Type  Name

   I*4  COMPRESS4                               FRCTOF                                  GET_QUEUE_INFO
        LINCHK                                  MOVC3

COMMAND QUALIFIERS

  FORTRAN /LIS=LIS$:LABEL/OBJ=OBJ$:LABEL MSRC$:LABEL

  /CHECK=(NOBOUNDS,OVERFLOW,NOUNDERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /STANDARD=(NOSYNTAX,NOSOURCE_FORM)
  /SHOW=(NOPREPROCESSOR,NOINCLUDE,MAP)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOCROSS_REFERENCE  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

Run Time:	7.35 seconds
Elapsed Time:	15.76 seconds
Page Faults:	182
Dynamic Memory:	208 pages

LPA11K
LIS

MCHECKS
LIS

LCPRINTER
LIS

MBA
LIS

LOGMSCP
LIS

LINCHK
LIS

MBAINT
LIS

LABEL
LIS

LOGGER
LIS

MASSTAPE
LIS

MASSDISK
LIS